

**Amendments to the Claims:**

This listing of claims replaces all prior versions, and listings, of claims in this application.

**Listing of Claims:**

1. (Currently Amended) A method for certifying software applications, said method comprising:

(a) creating a vulnerability knowledge database comprising one or more classes of known software vulnerabilities;

(b) applying a code parser to the software application to create an abstract syntax tree;

(c) comparing the abstract syntax tree and the classes of known software vulnerabilities to identify a set of potential exploitable software vulnerabilities;

(d) performing a static analysis of the source code, wherein the static analysis is a flow sensitive analysis of a list of constraints, wherein a constraint is a formal assertion describing how a program, function or procedure would affect a state of the software application if the software application were executed, and wherein the results of the static analysis comprise a set of exploitable software vulnerabilities;

(e) performing a first dynamic analysis of the software, wherein the first dynamic analysis comprises a set of tests to achieve code coverage;

(f) performing a second dynamic analysis of the software, wherein the second dynamic analysis comprises injecting faults into the software while being executed; ~~and~~

(g) performing any two of said analysis steps in a pipelined manner; and

(h) using the results of steps (a)-(g) as a basis for certifying the software application.

2. (Previously Presented) The method of claim 1, further comprising:  
performing a dynamic analysis of the set of exploitable software vulnerabilities to identify one or more false positives in the set of exploitable software vulnerabilities; and  
discarding the one or more false positives from the set of exploitable software vulnerabilities.

3. (Previously Presented) The method of claim 2, wherein performing the dynamic analysis comprises executing the set of potential exploitable software vulnerabilities with a maximal number of testing configurations.

4. (Previously Presented) The method of claim 1, wherein the vulnerability knowledge database is expandable.

5. (Previously Presented) The method of claim 1, wherein the set of exploitable software vulnerabilities comprises one or more of a security vulnerability, a safety vulnerability, or a reliability vulnerability.

6. (Currently Amended) A system for certifying a software applications, the system comprising:

a vulnerability knowledge database comprising one or more classes of known software vulnerabilities;

a code parser that creates an abstract syntax tree from the software application;

a vulnerability code analyzer that compares the abstract syntax tree the classes of known software vulnerabilities to identify a set of potential exploitable software vulnerabilities;

a static analysis tool that performs a static analysis of the source code, wherein the static analysis is flow sensitive analysis of a list of constraints, wherein a constraint is a formal assertion describing how a program, function or procedure would affect a state of the software application if the software application were executed, and wherein the results of the static analysis comprise a set of exploitable software vulnerabilities;

a first dynamic analysis tool that comprises a set of tests to achieve code coverage; and

a second dynamic analysis tool that operable to inject faults into the software while being executed,

wherein any two of said tools are accessed in a pipelined manner, and

wherein results generated by the system are used as a basis for certifying the software application.

7. (Previously Presented) The system of claim 6, further comprising a third dynamic analysis tool that performs a dynamic analysis of the set of exploitable software vulnerabilities to identify one or more false positives in the set of exploitable software vulnerabilities, wherein the one or more false positives are discarded from the set of exploitable software vulnerabilities.

8. (Previously Presented) The system of claim 7, wherein the third dynamic analysis tool executes the set of potential exploitable software vulnerabilities with a maximal number of testing configurations.

9. (Previously Presented) The system of claim 6, wherein the vulnerability knowledge database is expandable.

10. (Previously Presented) The system of claim 9, further comprising a user interface that enables a user to enter an additional known software vulnerability to the vulnerability knowledge database.

11. (Previously Presented) The system of claim 6, wherein the set of exploitable software vulnerabilities comprises one or more of a security vulnerability, a safety vulnerability, or a reliability vulnerability.

12-19. (Canceled)